Amendments to the Claims:

Please amend claims 1 and 12 as shown in the listing of claims below. This listing of claims will replace all prior versions and listings of claims in the application.

1. (currently amended) A method for generating program overlays from a sequence of program code, each overlay having a set of code and related data contained therein, the overlays being transferred via an overlay manager from a storage area to a receiving area for processing, the method comprising the steps of:

(a) breaking the sequence of program code into a set of segments, wherein each segment contains a certain amount of related code for processing;

(b) running a code segment in the set through a linker device;

(c) extracting the code segment and related data segment produced by the linker device, with each associated pair of code and data segments representing an overlay;

(d) checking if more segments exist in the set, if yes, then return to step (b), else proceed to step (e); and

(e) concatenating the ~~overlays into a file~~ associated code and data segments into paired sets which can be referenced by the overlay manager.

2. (original) The method according to Claim 1, wherein the step of breaking the sequence of program code into a set of segments includes dividing the code into a common code area, and an overlay code area.

3. (original) The method according to Claim 1, wherein the step of breaking the sequence of program code into a set of segments includes sizing the program code segments so that they will fit within the receiving area.

4. (original) The method according to Claim 1, wherein the steps further include creating stubs for referencing each function in each program code segment, the stubs being stored in the receiving area.

5. (original) The method according to Claim 4, wherein the steps further include generating an overlay table to be used in facilitating transfer of the overlays from the storage area to the receiving area, the overlay table being stored in the receiving area.

6. (original) The method according to Claim 1, wherein the storage area includes an external storage means.

7. (original) The method according to Claim 1, wherein the storage area includes memory associated with a low-MIPS processing device.

8. (original) The method according to Claim 1, wherein the receiving area includes memory associated with a high-MIPS processing device.

9. (original) The method according to Claim 8, wherein the high-MIPS processing device includes a digital signal processor.

10. (original) The method according to Claim 1, wherein after the concatenating step, the information is converted into a form usable by a processor.

11. (original) The method according to Claim 10, wherein the form includes a source file of a high-level programming language.

12. (currently amended) A method for generating program overlays from a sequence of program code, the program code having common code and code to be overlaid, each overlay having a set of code and related data contained therein, the overlays being transferred via an overlay manager from a storage area to a receiving area for processing, the method comprising the steps of:

(a) reserving a memory segment in the receiving area to hold overlaid code and data;

(b) breaking the sequence of code to be overlaid into a set of segments, wherein each segment contains a certain amount of related code for processing, and each segment is sized to fit in the reserved memory segment;

(c) creating stubs for each code segment, whereby the stubs represent entry points for functions within each code segment;

(d) linking the common code along with the stubs for each code segment;

(e) importing symbols from the common code and linking the next individual code segment in the set of segments to produce an image;

(f) extracting overlay code and data from the image produced in step (e);

(g) checking if more segments exist in the set, if yes, then return to step (e), else proceed to step (h); and

(h) concatenating the ~~overlays into a file~~ associated code and data segments into paired sets which can be referenced by the overlay manager.

13. (original) The method according to Claim 12, wherein the storage area includes an external storage means.

14. (original) The method according to Claim 12, wherein the storage area includes memory associated with a low-MIPS processing device.

15. (original) The method according to Claim 12, wherein the receiving area includes memory associate with a high-MIPS processing device.

16. (original) The method according to Claim 15, wherein the high-MIPS processing device includes a digital signal processor.

17. (original) The method according to Claim 12, wherein after the concatenating step, the information is converted into a form usable by a processor.

18. (original) The method according to Claim 17, wherein the form includes a source file of a high-level programming language.

19. (original) A method for generating program overlays from a sequence of program code, the program code having common code area and overlay code area, each overlay having a set of code and related data contained therein, the overlays being transferred via an overlay manager from a storage area to a receiving area for processing, the method comprising the steps of:

(a) analyzing the overlay code area and determining the function entry points for each overlay;

(b) creating an overlay control file for each overlay, whereby the overlay control file describes each pair of code and data associated with each overlay;

(c) generating a wrapper file from the overlay control file;

(d) creating a linker command file for the common area;

(e) creating a linker command file for the overlay area;

(f) performing an initialization for the overlay;

(g) creating a common image for the code and data;

(h) producing overlay sections from the image;

(i) producing an overlay sections file; and

(j) producing a load command file, whereby the command file will load the overlay sections file into the appropriate receiving area.

20. (original) The method of Claim 19, wherein the step of producing overlay sections from the image includes the following steps:

(a) creating a copy of the common image, whereby the entry point symbols are removed from the particular overlay to be built;

(b) linking together an image for a particular overlay to form an overlay image file; and

(c) extracting the code and data sections from the overlay image file.

21. (original) The method of Claim 19, wherein the step of generating a wrapper file reads the overlay control file and generates wrapper functions for each function described therein

22. (previously presented) The method of Claim 21, wherein the wrapper file includes:

an overlay descriptor, which resides in common data and contains information about the overlay;

the wrapper function, which is the entry point to the overlay function; and

a fault function, which causes the overlay code and data sections to be paged from the storage area to the receiving area.

23. (original) The method according to Claim 19, wherein the storage area includes an external storage means.

24. (original) The method according to Claim 19, wherein the storage area includes memory associated with a low-MIPS processing device.

25. (original) The method according to Claim 19, wherein the receiving area includes memory associated with a high-MIPS processing device.

26. (original) The method according to Claim 25, wherein the high-MIPS processing device includes a digital signal processor.

27. (original) The method according to Claim 19, wherein after the concatenating step, the information is converted into a form usable by a processor.

28. (original) The method according to Claim 27, wherein the form includes a source file of a high-level programming language.